Adv Syst Sci Appl 2024; **02**:94–102
Published online at https://ijassa.ipu.ru.

Practical Applicability of the Metric Approach for a Scheduling Problem

Alexander Lazarev[1*], Darya Lemtyuzhnikova[1,2], Ilja Kudinov[1]

[1] *V.A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia,*
[2] *Moscow Aviation Institute, Moscow, Russia*

Abstract: A given special case of NP-complete scheduling problem can be approximated by solving a special case of similar problem with the same precedence graph. We construct a metric space over a set of special cases of this problem and consider the statistical relationship between distance between a pair of special cases of the under consideration and the average error of the approximated solution. Sethi, Gabow, Coffman's and Fujii's algorithms for this problem are used. It is shown that the absolute and the relative error of the objective function decreases over the density of a graph with a fixed number of jobs. In general case, relative non-zero error value increases with the number of jobs.

Keywords: Scheduling, Optimization and Control, Operations Research, Makespan

1. INTRODUCTION

Nowadays various polynomial algorithms for many combinatoric optimization problems have been developed, but there remain many of them for which the question of existence of polynomial algorithm is still open. At the same time absolute approximate algorithms are known for very few NP-hard optimization problems and in most cases only relative error of objective function value estimates are available. Therefore, it is important to develop approximate algorithms that allow to obtain solutions of the problem with an acceptable error estimation.

The metric approach as one of the effective modern approach is considered in this paper. The main idea of this is following. Assume a metric ρ between a pair of instances A, B of some problem:

$$\rho(A, B) \geq f^A(x_B^*) - f^A(x_A^*), \tag{1.1}$$

where $f^A(x)$ is objective function values of instance A, and the arguments x_A^* and x_B^* are an optimal solution of instances A and B corresponding. Then, the metric become an upper bound of the absolute error of using the optimal solution x_B^* of instance B as an approximate solution of instance A. If the searching for an optimal solution of instance A is significantly more time-consuming compared to finding that of B, and we know a rather small upper bound of $\rho(A, B)$, we can use x_B^* as a approximate solution of instance A.

In this paper we investigate the feasibility and practicability of applying the metric approach to solve the NP-complete scheduling problem of processing jobs on two parallel machines with a partial ordered set of jobs: $P2 \mid prec,\ p_j \in \{1, 2\} \mid C_{max}$, where processing times of jobs equal 1 or 2.

[*] Corresponding author: Alexander Lazarev, jobmath@mail.ru

There exist a little of experiments conducted as a proof of method applicability to problems of different types. Our research is based on Lazarev's papers [1] and [2], in which the way of metric's construction for multiple problems is discussed. Moreover, we conducted an experiment for a single machine scheduling problem.

To fulfill the existing knowledge of feasibility to apply metric approach for various types of problems, we conduct experiments for problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$. We chose this simplification of the problem $P2 \mid prec,\ p_j = k \mid C_{max}$ for conduction the experiments due to the existence of the variety of polynomial algorithms for the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$, to which we can easily apply metric approach. Nevertheless, we apply the proof of metric construction for the problem $P2 \mid prec,\ p_j = k \mid C_{max}$, where $k \in \mathbb{Z}^+$.

One of the first papers concerning an algorithm for optimal solution obtaining for problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$ is the work of Fujii [4], in which the algorithm based on finding the maximum matching in precedence graph was proposed. The upper bound of the number of operations required to find the optimal sequence is $O(n^3)$ operations where n is number of jobs.

The Coffman [5] and Sethi [6] proposed algorithms which, as the Fujii's algorithm, are based on a list of jobs, the list is used to sequentially take elements for distribution on two machines. Coffman's algorithm has found the upper bound is $O(n^2)$ operations, and Sethi's work presents two algorithms, one of which makes the labeling for $O(e + n)$ operations, and the other makes the schedule for $O(e + n\alpha(n))$ operations, where e – the number of edges in the precedence graph, and $\alpha(n)$ – Ackerman's function, which slowly grows. The labeling algorithm proposed by Sethi is largely based on Coffman's algorithm, but it adds an additional examination of the structure at each level of the graph.

Gabow's algorithm [7] works according to the "high level first" (HLF) principle and requires $O(e + n\alpha(n))$ operations.

The NP-completeness of the problems $P2 \mid prec,\ p_j = 1 \mid C_{max}$ and $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ was first proved Ullman [8]. Furthermore, the Bevern in paper [9] proves that the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ is $W[2]$-hard parameterized by the width of the partial order. However it is an open question if $Pm \mid prec, p_j = 1 \mid C_{max}$ is NP-hard for $m \geqslant 3$.

In literature there are references to the work of [10], which gives an algorithm whose complexity is $O(n \log n)$ operations for the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$, and a heuristic algorithm [11], in the worst case giving error equal to one for the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$. In addition, in [12] an algorithm is given that yields the optimal solution for the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ using $O(n^2 \log n)$ operations. Hu [13] demonstrates polynomial algorithm for the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$, if graph of ordering restrictions is a tree and number of jobs satisfies some conditions.

By the way, Dolev's methods [14] lead to polynomial algorithms for the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$ if the number of machines is fixed and the precedence graph has a certain form. In particular, if the precedence graph contains only in-trees or out-trees, the result leads to linear algorithms for finding an optimal schedule on two and three machines.

The structure of the paper is organized as follows. In Section 2 the mathematical formulation of the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ is given. In Section 3 we define the metric we use. In Section 4 we describe our motivation and approach for the problem. In Section 5 the experiment's result is given.

2. THE PROBLEM DEFINITION

We consider scheduling the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$. Each instance A has a set of n jobs $N^A = N$ and set of m machines $M^A = M$. Each job $j \in N$ can be processed on one machine during p_j time units. Each machine can process only one job at a time. Set

N has a partial order $\prec$ that from $j_1 \prec j_2$ follows j_2 can not be started processing until j_1 be complete. The partial order is defined by a dag (Directed Acyclic Graph) called a precedence graph.

We denote $\pi^A = \pi$ as a permutation $(j_1,\ j_2,\ \ldots,\ j_n)$ over the job set N. In order of constructing a schedule we put each job from π sequentially to the earliest possible place of processing in the schedule. For simplicity, we call π a schedule as well. If $s_j^A(\pi)$ is a starting time of job $j \in N$ processing and $C_j^A(\pi) = s_j^A(\pi) + p_j$ is the completion one, the schedule π has the following features:

1. If $m_{j_1} = m_{j_2}$ then $[s_{j_1}(\pi),\, C_{j_1}(\pi)) \cap [s_{j_2}(\pi),\, C_{j_2}(\pi)) = \emptyset$ for $j_1, j_2 \in N$.
2. If $j_1 \prec j_2$ then $C_{j_1}(\pi) \le s_{j_2}(\pi)$.

The first condition means that a machine cannot process more than one job at a time. The second condition is nothing more than the fulfilltime of precedence relations between jobs.

A schedule π of an instance A is feasible if it does not disrupt the conditions above. One schedule π can be feasible for a number of instances with the same precedence graphs.

As is usually denoted in scheduling theory, we will denote makespane by $C_{max}(\pi)$:

$$C_{max}(\pi) = \max_{j \in N} C_j(\pi).$$

A schedule π^* is called an optimal schedule for an instance A of the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ if $C_{max}(\pi^*) \le C_{max}(\pi)$ for each feasible π.

As indicated [9], the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ is NP-complete with respect to the width of the partial order. Therefore, this is the reason why approximate solutions of the problem are considered.

3. METRIC OVER THE SET OF INSTANCES OF THE PROBLEM $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$

We consider the metric approach application for the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$. Let π_A^* be an optimal solution of instance A, and π_B^* is that of instance B. Using C_{max} instead of f into (1.1) gives us, the following:

$$\rho(A,B) \ge C_{max}^A(\pi_B^*) - C_{max}^A(\pi_A^*). \tag{3.2}$$

There is a metric for the problem $P2 \mid p_j \mid C_{max}$ [3]:

$$\rho(A,B) = \sum_{j \in N} |p_j^A - p_j^B|, \tag{3.3}$$

where instances A and B have the same number of jobs n. This is Minkowski of order 1. In case of the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$ there is no known metric for the problem with partial ordered over N. So we believe that for pairs of instances A and B that do not differ otherwise than a partial order over N the metric takes the minimal solution only when the partial order of both A and B is the same.

Lemma 3.1:
Let A and B be instances of the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$, and π_A^*, π_B^* are optimal solutions of A, B corresponding. Then:

$$C_{max}^A(\pi_B^*) - C_{max}^A(\pi_A^*) \le \sum_{j \in N} |p_j^A - p_j^B|. \tag{3.4}$$

Proof
Let's consider the following difference:

$$|C^A_{max}(\pi) - C^B_{max}(\pi)| \tag{3.5}$$

for any A, B, π and give an upper bound of this.

Now we prove that the maximum of (3.5) reaches on π on which every job is processed on only one machine, so the other one processes no one job. Obviously, the decrease value of the target function value is not greater than the difference between sums of processing times of all jobs in terms of each instances A and B:

$$|C^A_{max}(\pi) - C^B_{max}(\pi)| \leq \left| \sum_{j \in N} p^A_j - \sum_{j \in N} p^B_j \right|.$$

Using the following commonly known inequality:

$$|a + b| \leq |a| + |b|,$$

we obtain

$$|C^A_{max}(\pi) - C^B_{max}(\pi)| \leq \left| \sum_{j \in N} p^A_j - \sum_{j \in N} p^B_j \right| \leq \sum_{j \in N} |p^A_j - p^B_j|. \tag{3.6}$$

Finally, using (3.6), we get

$$\begin{aligned} &C^A_{max}(\pi^*_B) - C^A_{max}(\pi^*_A) = \\ &= \left(C^A_{max}(\pi^*_B) - C^B_{max}(\pi^*_B)\right) + \left(C^B_{max}(\pi^*_B) - C^B_{max}(\pi^*_A)\right) + \left(C^B_{max}(\pi^*_A) - C^A_{max}(\pi^*_A)\right) \leq \\ &\leq \left(C^B_{max}(\pi^*_B) - C^B_{max}(\pi^*_A)\right) + 2 \sum_{j \in N} |p^A_j - p^B_j|. \end{aligned}$$

That is

$$\left(C^A_{max}(\pi^*_B) - C^A_{max}(\pi^*_A)\right) + \left(C^B_{max}(\pi^*_B) - C^B_{max}(\pi^*_A)\right) \leq 2 \sum_{j \in N} |p^A_j - p^B_j|.$$

From symmetry considerations, we finally obtain the following result:

$$C^A_{max}(\pi^*_B) - C^A_{max}(\pi^*_A) \leq \sum_{j \in N} |p^A_j - p^B_j|.$$

□

Theorem 3.1:
The function ρ from (3.3) is a metric over the sets of instances of the problem $P2 \mid prec,\ p_j \in \{1, 2\} \mid C_{max}$.

Proof
That (3.4) corresponds to the axioms of the metric is obvious:

1. $\rho(A, B) = 0 \iff A = B$:

$$\rho(A, B) = \sum_{j \in N} |p^A_j - p^B_j| = 0 \iff p^A_j = p^B_j,\ \forall j \in N.$$

2. $\rho(A, B) \geq 0$:
$$\rho(A, B) = \sum_{j \in N} |p_j^A - p_j^B| \geq 0.$$
3. $\rho(A, B) = \rho(B, A)$:
$$\rho(A, B) = \sum_{j \in N} |p_j^A - p_j^B| = \sum_{j \in N} | - p_j^A + p_j^B| = \rho(B, A) =$$
4. $\rho(A, C) \leq \rho(A, B) + \rho(B, C)$: since $|a + b| \leq |a| + |b|$,
$$\rho(A, C) = \sum_{j \in N} |p_j^A - p_j^C| = \sum_{j \in N} |p_j^A - p_j^B + p_j^B - p_j^C| \leq \sum_{j \in N} |p_j^A - p_j^B| + \sum_{j \in N} |p_j^B - p_j^C| = \rho(A, B) + \rho(B, C).$$

□

4. THE POLYNOMIAL-TIME APPROXIMATION SCHEME

As stated above, an instance of the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$ can be solved by polynomial algorithms. It is possible to convert an instance A of the problem $P2 \mid prec,\ p_j \in \{1, 2\} \mid C_{max}$ to an instance B of the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$ by setting $p_j = 1$ for all $j \in N$. This leads to a polynomial-time approximation scheme (PTAS) that can be seen on Fig. 4.1.

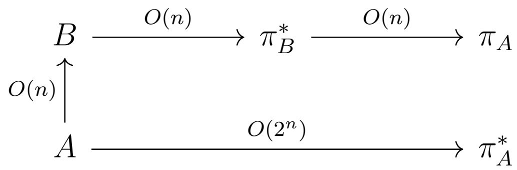


Fig. 4.1. The PTAS scheme for the problem $P2 \mid prec,\ p_j \in \{1, 2\} \mid C_{max}$.

An instance of the problem $P2 \mid prec,\ p_j \in \{1, 2\} \mid C_{max}$ has a set of feasible schedules. But some of them differs from the other only by the order of jobs on each machine, not by the makespan value. Since we aim to minimize target function C_{max}, we take only those schedules that have the minimal value of C_{max}.

No solving instance B algorithm from the ones above must be better than others in general case. Indeed, let $\mathcal{F}^*(A) \subseteq \mathcal{F}(A)$ be a subset of optimal schedules from $\mathcal{F}(A)$. Since $G(A) = G(B)$, the sets $\mathcal{F}(A)$ and $\mathcal{F}(B)$ are equal too. But sets $\mathcal{F}^*(A)$ and $\mathcal{F}^*(B)$ are completely different in general case. It means that the polynomial algorithms one use for the instance B of the problem $P2 \mid prec,\ p_j = 1 \mid C_{max}$ generally just take a random schedule from $\mathcal{F}(A)$ as an approximate solution π_A for the instance A.

5. COMPUTER EXPERIMENTS

Four polynomial algorithms for the problem $P2 \mid prec, p_j = 1 \mid C_{max}$: Fujii's Algorithm [4], Coffman's Algorithm [5], Sethi's Algorithm [6] and Gabow's Algorithm [7] were implemented. Optimal solution of NP-hard problem $P2 \mid prec, p_j \in \{1,2\} \mid C_{max}$ was obtained by GLPK solver[†].

The following indicators were chosen as efficiency evaluation criteria:

1. μ determines the share of cases for which no optimal solution was obtained by the approximate algorithm:
$$\mu = \frac{\bar{K}}{K},$$
where K is the number of generated examples, and $\bar{K}$ is the one for which the found solution was not optimal, i.e:
$$C_{max}(\pi_i) - C_{max}(\pi_i^*) > 0;$$
2. Average relative non-zero error β_{nre}:
$$\beta_{nre} = \frac{1}{\bar{K}} \sum_{i=1}^{\bar{K}} \frac{C_{max}(\pi_i) - C_{max}(\pi_i^*)}{C_{max}(\pi_i^*)};$$
3. Average absolute non-zero error β_{nae}:
$$\beta_{nae} = \frac{1}{\bar{K}} \sum_{i=1}^{\bar{K}} \left(C_{max}(\pi_{i,A}) - C_{max}(\pi_{i,A}^*)\right);$$
4. Average relative error β_{re}:
$$\beta_{re} = \frac{1}{K} \sum_{i=1}^{K} \frac{C_{max}(\pi_i) - C_{max}(\pi_i^*)}{C_{max}(\pi_i^*)};$$
5. Average absolute error β_{ae}:
$$\beta_{ae} = \frac{1}{K} \sum_{i=1}^{K} \left(C_{max}(\pi_{i,A}) - C_{max}(\pi_{i,A}^*)\right).$$

There were calculated 1000 of uniformly distributed random instances of the problem $P2 \mid prec, p_j = \{1,2\} \mid C_{max}$ for instance with number of jobs $N \in \{3,4,\ldots,13\}$ with the density $D = 0.3$, where the density is the ratio of the number of edges of the given graph to the one of the complete graph, and 250 of uniformly distributed random instances for each problem for density values $D \in \{0.1, 0.2, \ldots, 1.0\}$ for problems with number of jobs $N \in \{3,4,\ldots,13\}$.

As can be seen from Fig. 5.2, the share of cases for which the algorithm obtained a non-optimal solution, μ grows no more than linearly with increasing the number of vertices (jobs) n. The fluctuations in the graph at odd n are caused by the fact that there are two machines in the problem $P2 \mid prec,\ p_j \in \{1,2\} \mid C_{max}$. When a number of jobs n is odd, on average the last job can be put into the schedule, leaving an empty space on another machine.

An important result is the dependence of the relative non-zero error β_{nre} and the absolute non-zero error β_{nae} on the number of vertices n. As can be seen from Fig. 5.3, respectively, β_{nre} decreases with increasing number of vertices, and β_{nae} grows slowly, which makes it possible to apply the method to graphs with a large number of vertices. The absolute error β_{nae} remains approximately at the same level for all n. This is a consequence of the fact that

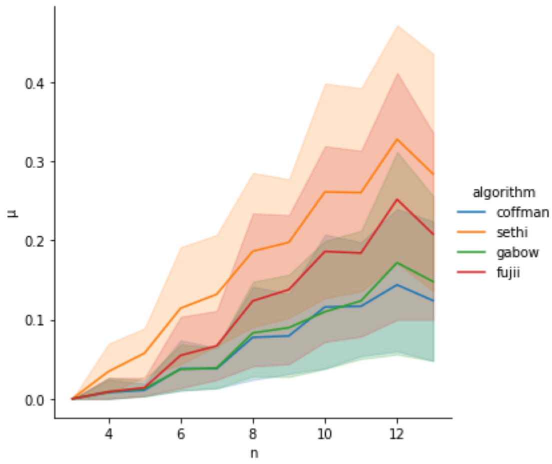
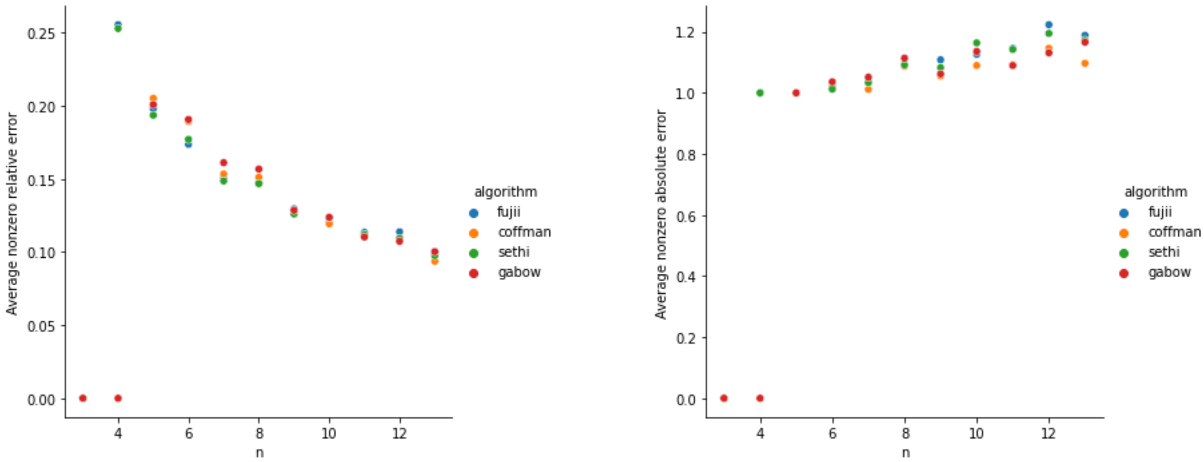


Fig. 5.2. Graph of the dependence of the share of cases μ for which the algorithm obtained a non-optimal solution on the number of vertices n. It slowly grows with the growth of n. Fluctuations between even and odd values of n is due to the fact that there are two machines in considered problem



(a) (b)

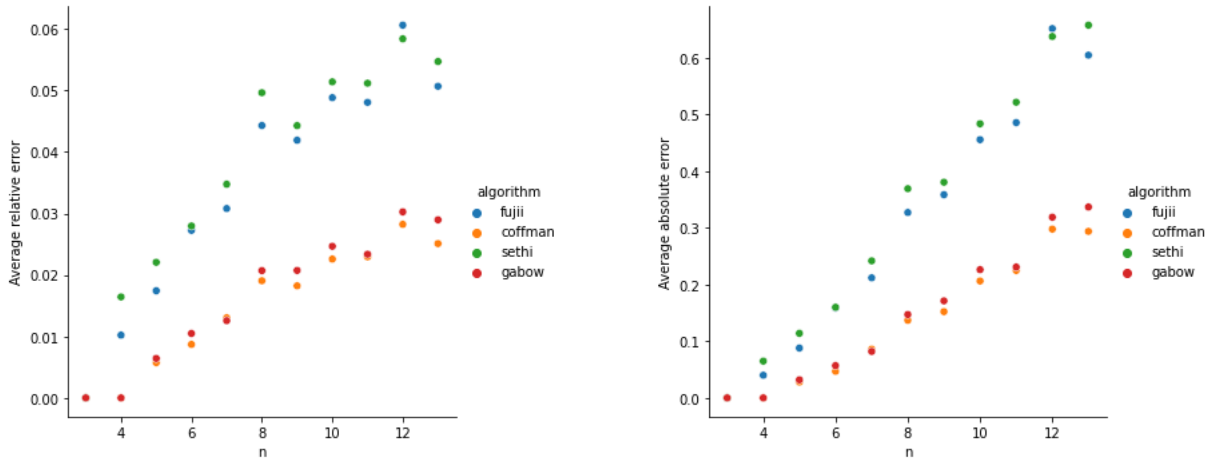
Fig. 5.3. Graphs of the dependence of the relative non-zero error β_{nre} (a) and the absolute non-zero error β_{nae} (b) indicators on the number of vertices n at 1000 generations. The cases $n \leq 4$ are trivial, and the polynomial algorithms always get an optimal solution

the polynomial algorithms absolute errors do not tend to accumulate overtime and ”fixes” each other due to the presence of two machines.

But if we start count cases of zero error, the full picture changes. As can be seen from Fig. 5.4, the graphs of relative and absolute errors on the number of vertices show a pairwise relationship between the algorithms. During the near-constant graph of β_{nae}, the growth of β_{ae} means growth of the rate of errors.

All indicators show a decrease in the value of the error when the density of the graph D increases, as look in Fig. 5.5. It seems that the less the graph density D, the smaller a space of feasible schedules, and the remain are closer to an optimal one.

† https://www.gnu.org/software/glpk/glpk.html



(a) (b)

Fig. 5.4. Graph of the dependence of the relative error β_{re} (a) and the absolute error β_{ae} (b) on the number of vertices n at 1000 generations

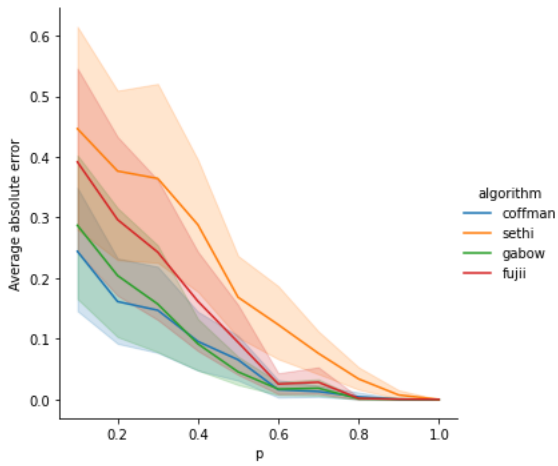


Fig. 5.5. Graph of the dependence of the absolute error β_{ae} on the density D of the graph at 250 generations

6. CONCLUSIONS

In this paper we consider the possibility of applying the metric approach to solve the NP-complete problem $P2 \mid prec, \ p_j \in \{1, 2\} \mid C_{max}$. It was shown that if the number of restrictions increases, the absolute and relative error of C_{max} change slowly. Besides, the relative non-zero error value decreases, if the number of vertices in the graph grows. In addition, it turned out to be more efficient to use the Coffman's and Gabow's algorithms in finding an approximate solution of this problem by the method discussed in this paper.

The obtained average absolute error $C_{max}(\pi_A) - C_{max}(\pi_A^*)$ seems to be depends only on n and D on sets of all dags of classes considered above. However, this result could not be proved analytically at this moment.

ACKNOWLEDGEMENTS

The results of this paper partly were obtained within the RSF grant (project No. 22-71-10131).

We thanks E. Bukueva for assistance on in the early stages of the work.

REFERENCES

1. Lazarev, A. (2019) *Scheduling theory. Methods and algorithms*. Moscow, Russia: ICS RAS.
2. Lazarev, A., Lemtyuzhnikova, D. & Werner F. (2021) A metric approach for scheduling problems with minimizing the maximum penalty, *Applied Mathematical Modelling*, **89**, 1163–1176.
3. Lazarev, A. & Arkhipov, B. (2018) Estimation of the Absolute Error and Polynomial Solvability for a Classical NP-Hard Scheduling Problem, *Doklady Mathematics*, **97**, 262–265.
4. Fujii, M., Kasami, T. & Ninomiya, K. (1969) Optimal sequencing of two equivalent processors, *SIAM Journal on Applied Mathematics*, **17**(4), 784–789.
5. Coffman E. G. & Graham R. L. (1972) Optimal scheduling for two-processor systems, *Acta informatica*, **1**(3), 200–213.
6. Sethi, R. (1976) Scheduling graphs on two processors, *SIAM Journal on Computing*, **5**(1), 73–82.
7. Gabow, H. N. (1982) An almost-linear algorithm for two-processor scheduling, *Journal of the ACM (JACM)*, **29**(3), 766–780.
8. Ullman, J. D. (1975) NP-complete scheduling problems, *Journal of Computer and System sciences*, **10**(3), 384–393.
9. van Bevern, R., Bredereck, R., Bulteau, L., Komusiewicz, C., Talmon, N., et al. (2016) Precedence-constrained scheduling problems parameterized by partial order width, *International conference on discrete optimization and operations research – Springer, Cham*, 105–120.
10. Nakajima, K., Leung, J. & Hakimi, S. L. (1981) Optimal two processor scheduling of tree precedence constrained tasks with two execution times, *Performance Evaluation*, **1**(4), 320–330.
11. Kaufman, M. T. (1974) An almost-optimal algorithm for the assembly line scheduling problem, *IEEE Transactions on Computers*, **100**(11), 1169–1174.
12. Du J. & Leung J. Y. T. (1989) Scheduling tree-structured tasks on two processors to minimize schedule length, *SIAM journal on discrete mathematics*, **2**(2), 176–196.
13. Hu, T. C. (1961) Parallel sequencing and assembly line problems, *Operations research*, **9**(6), 841–848.
14. Dolev, D. & Warmuth, M. (1985) Scheduling flat graphs, *SIAM Journal on Computing*, **14**(3), 638–657.

